EXHIBIT W

FILED UNDER SEAL

CLEMENT SETH ROBERTS (STATE BAR NO. 209203)
croberts@orrick.com
BAS DE BLANK (STATE BAR NO. 191487)
basdeblank@orrick.com
ALYSSA CARIDIS (STATE BAR NO. 260103)
acaridis@orrick.com
EVAN D. BREWER (STATE BAR NO. 304411)
ebrewer@orrick.com
ORRICK, HERRINGTON & SUTCLIFFE LLP
The Orrick Building
405 Howard Street
San Francisco, CA 94105-2669
Telephone: +1 415 773 5700
Facsimile: +1 415 773 5759

SEAN M. SULLIVAN (*pro hac* vice)
sullivan@ls3ip.com
MICHAEL P. BOYEA (*pro hac* vice)
boyea@ls3ip.com
COLE B. RICHTER (*pro hac* vice)
richter@ls3ip.com
LEE SULLIVAN SHEA & SMITH LLP
656 W Randolph St., Floor 5W
Chicago, IL 60661
Telephone: +1 312 754 0002
Facsimile: +1 312 754 0003

Attorneys for Sonos, Inc.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

GOOGLE LLC, Plaintiff and Counter-defendant, v. SONOS, INC., Defendant and Counter-claimant.	Case No. 3:20-cv-06754-WHA Related to Case No. 3:21-cv-07559-WHA **REPLY EXPERT REPORT OF DOUGLAS C. SCHMIDT**

available, acceptable non-infringing alternative at the time of Google's first infringement for at least the reasons discussed above and discussed in my Prior Submissions.

B. Dr. Bhattacharjee's Onesie and/or Streaming Watch Alternative

259. Dr. Bhattacharjee opines that "as [he] showed in [his] opening report, a non-infringing alternative that was available to Google at the time the '033 patent issued was for the receiver device not to send the accused getWatchNext and/or getPlayer requests." Bhatta. Rebuttal Report, ¶288. I disagree for various reasons.

260. To start, Google and Dr. Bhattacharjee's explanation of this purported alternative has been a moving target, ambiguous as to what the alleged alternative even is, and inconsistent with respect to what functionality is involved.

261. As I explained in my Opening Report, Google's own explanation of this purported alternative to the Asserted Claims of the '033 Patent is vague and incomplete. *See* Schmidt Op. Report, ¶¶496-504. I have been informed that Google first identified and explained this purported alternative as to the Asserted Claims of the '033 Patent on November 21, 2022, which I understand was a mere 9 days before fact discovery closed on November 30, 2022. *See* Google LLC's Eighth Supplemental Objections and Responses to Plaintiff Sonos Inc.'s First Set of Fact Discovery Interrogatories (No. 18), pp. 13-14. In any case, Google's interrogatory response states the following:

> [A] non-infringing alternative is for the receiver device to not send a GetWatchNext message or getPlayer requests. Instead, in Alternative 32 [*sic*] the receiver device would send a request to a Onesie agent. The Onesie agent would send the GetWatchNext request and GetPlayer request, and then stream the media data from the Onesie agent to the receiver device. In other words, in Alternative #3 the receiver device does not "obtain data identifying a next one or more media items that are in the remote playback queue," and does not "use the obtained data to retrieve at least one media item in the remote playback queue from the cloud-based media service."

Id., p. 13.

262. Google continues by asserting that it "implemented Onesie for many receiver devices and is currently working on a feature called Streaming Watch which can allow the Onesie agent to make the GetWatchNext and GetPlayer calls." *Id.*, pp. 13-14. Google concludes by stating "Onesie and Streaming watch are discussed in further detail in the opening and rebuttal [showdown] reports of Dr. Bhattacharjee, which are incorporated herein by reference." *Id.*, p. 14.

263. While Dr. Bhattacharjee did discuss Onesie and Streaming Watch in this Showdown reports, that discussion was in the context of claim 13 of the '615 Patent that has different limitations than the Asserted Claims of '033 Patent. In that context, with respect to Onesie, Dr. Bhattacharjee stated that, for a non-Casting use case, "Google has implemented its 'Onesie' platform in which after receiving a setPlaylist message a YouTube receiver sends a single request to a Onesie Agent to obtain both the alleged 'resource locators' and multimedia content." Bhatta. Op. Showdown Report, ¶590. Dr. Bhattacharjee characterized "the alleged 'resource locators'" as "a WatchNext message, or streaming URL from the Player service …." *Id.*, ¶589. As I explained before, a WatchNext response that a Receiver obtains contains a videoId identifying the next media item that the Receiver is to playback.

264. Dr. Bhattacharjee continued by stating he understood that:

> Google is also working on a feature called Streaming Watch, which would further allow the Onesie agent to provide ***the information in a WatchNext response***: "The idea here is that clients make a single request to Onesie. Onesie will then make the GetPlayer and GetWatchNext RPCs. Onesie can then stream back to the client the GetPlayer response ***ahead of the GetWatchNext response*** via chunked HTTP." GOOG-SONOSNDCA-00071671[.]

Bhatta. Op. Showdown Report, ¶592.

265. What's notable here is that Dr. Bhattacharjee explained that, in the purported combined Onesie/Streaming-Watch alternative, the Receiver would obtain a GetPlayer response for the current media item is to playback, as well as a WatchNext response that would contain a videoId of the next media item that the Receiver is to playback. Indeed, Dr. Bhattacharjee stated the following in his Rebuttal Showdown Report regarding the purported combined Onesie/Streaming-Watch alternative: "[p]ut another way, with Streaming Watch Google's NIA No. 1 would merge the functionality of the WatchNext and Bandaid servers, such that the WatchNext servers would not be a separate set of servers, as required by Claim 13." Bhatta. Op. Showdown Report, ¶361; *see also, id.*, ¶364 ("Onesie alone (without using Streaming Watch) would serve URLs "from the same server and/or service as the content"). Streaming Watch is directed only at Sonos's argument that a videoId sent to the receiver by a WatchNext service is a resource locator.").

266. In sum, in his Showdown Reports for the '615 Patent, Dr. Bhattacharjee explained

that the purported combined Onesie/Streaming-Watch alternative involved a Receiver making a request to Onesie with a videoId of a current media item[11] and the Receiver obtaining a GetPlayer response for the current media item and a WatchNext response that would contain a videoId of the next media item.

267. Turning to Dr. Bhattacharjee's Opening Report for the '033 Patent, Dr. Bhattacharjee's opinions and analysis of this alleged alternative were likewise vague and incomplete to such an extent that it appeared that Dr. Bhattacharjee did not firmly grasp what Google is even proposing, as I explained in my Rebuttal Report. *See* Schmidt Rebuttal Report, ¶1010. In fact, Dr. Bhattacharjee's Opening Report essentially just parroted back verbatim what Google set forth in its November 21 interrogatory response and referred back to Dr. Bhattacharjee's Showdown Reports:

Google Rog. No. 18 Response	**Bhatta. Op. Report, ¶¶759, 761**
[A] non-infringing alternative is for the receiver device to not send a GetWatchNext message or getPlayer requests. Instead, in Alternative 32 [sic] the receiver device would send a request to a Onesie agent. The Onesie agent would send the GetWatchNext request and GetPlayer request, and then stream the media data from the Onesie agent to the receiver device. In other words, in Alternative #3 the receiver device does not "obtain data identifying a next one or more media items that are in the remote playback queue," and does not "use the obtained data to retrieve at least one media item in the remote playback queue from the cloud-based media service." Google has implemented Onesie for many receiver devices and is currently working on a feature called Streaming Watch which can allow the Onesie agent to make the GetWatchNext and GetPlayer calls. See, e.g., GOOG-SONOSNDCA-00071671; GOOG-SONOSNDCA-00070863; GOOG-	[A] non-infringing alternative is for the receiver device to not send a GetWatchNext message or getPlayer requests. Instead, in Alternative #2 the receiver device would send a request to a Onesie agent. The Onesie agent would send the GetWatchNext request and GetPlayer request, and then stream the media data from the Onesie agent to the receiver device. In other words, in Alternative#2 the receiver device does not "obtain data identifying a next one or more media items that are in the remote playback queue," and does not "use the obtained data to retrieve at least one media item in the remote playback queue from the cloud-based media service." I understand that Google has implemented Onesie for many receiver devices and is currently working on a feature called Streaming Watch which can allow the Onesie agent to make the GetWatchNext as well as the GetPlayer call. *See, e.g.*, GOOG-SONOSNDCA-00071671; GOOG-

[11] *See, e.g.*, Bhatta. Rebuttal Report, ¶¶289 ("[W]ith the now launched Onesie feature the request from the receiver to the Onesie agent would include the videoId and/or playlist ID for the Shared Queue."), 297 ("[A] videoId is used to retrieve the first several seconds of media content. *See also*, *e.g.*, GOOG-SONOSNDCA-00073494, 95 ('Onesie only streams the first seven seconds of a video. For the rest of the video, the Player needs to fetch it from the content-hosting machine.').")

SONOSNDCA-00073429; GOOG-SONOSNDCA-00073577. Onesie and Streaming watch are discussed in further detail in the opening and rebuttal reports of Dr. Bhattacharjee, which are incorporated herein by reference.	SONOSNDCA-00070863; GOOG-SONOSNDCA-00073429; GOOG-SONOSNDCA- 00073577. Onesie and StreamingWatch are discussed in greater detail in my opening and rebuttal reports for claim 13 of the '615 patent, which I incorporate herein by reference.

268. Now, in his Rebuttal Report, Dr. Bhattacharjee describes the purported combined Onesie/Streaming-Watch alternative in much greater detail than he or Google has ever before, but some of his new description is inconsistent with his description in his Showdown Reports. *See* Bhatta. Rebuttal Report, ¶¶289, 295-97,

269. For example, Dr. Bhattacharjee now claims that the request that the Receiver sends to the Onesie agent no longer includes a videoId but rather only a playlistId. *See* Bhatta. Rebuttal Report, ¶289 ("[I]n the Streaming Watch alternative [he] discuss[es] below the request to the Onesie agent would include a playlist ID and any associated servers could then determine the starting video from the playlist ID and stream the content for the starting video and subsequent videos in the playlist to the receiver[.]"). I note that the very passage from a Google internal document that Dr. Bhattacharjee cites for this proposition states that the request "***could*** just include a playlist ID," which means that this appears to just be a brainstorming exercise by Google's engineers, but more to the point, the passage explains that the discussion of only including a playlistId to the exclusion of a videoId in the request is only for "certain types of playlists (shuffle, radio)." *Id.* Thus, it does not appear to me that (i) Dr. Bhattacharjee is accurately describing the contents of the request that the Receiver sends and (ii) the contents of the request is the same for all types of playback.

270. As another example, Dr. Bhattacharjee explains that, for the GetPlayer aspect of this purported alternative, a Receiver "would send a Onesie request for a media item (e.g., a song or video) each time it starts playback of the next media item" and "the Onesie response returns media content for the beginning of the song or video … followed by the 'Player Response' containing the Bandaid URLs that the receiver device can use to request the remaining chunks of media content" and he opines "[t]he Bandaid URLs returned in a 'Player Response' of the Onesie request are not 'data identifying a next one or more media items that are in the [alleged] remote playback queue[.]'"

Bhatta. Rebuttal Report, ¶296.

271. Having the benefit of more details of this purported combined Onesie/Streaming-Watch alternative, I agree that a Bandaid URL does not amount to limitation 1.7's "data identifying a next one or more media items that are in the remote playback queue," but as I explain below, ***the videoId*** for the next media item contained in the WatchNextResponse portion of the Onesie response does. Indeed, Dr. Bhattacharjee acknowledges that, in this purported combined Onesie/Streaming-Watch alternative, "a videoId is used to retrieve the first several seconds of media content." Bhatta. Rebuttal Report, ¶297. As I explained above, I disagree with Dr. Bhattacharjee's position that limitation 1.7 requires a single piece of data to be used to retrieve the ***entire*** media item, but rather, limitation 1.7 merely requires the "playback device" be configured to "***use the obtained data to retrieve*** at least one media item in the remote playback queue from the cloud-based media service," which the Receiver does with the videoId of the next media item. *Supra* ¶¶189-94.

272. As another example, in response to my explanation "that this non-infringing alternative would still infringe because … a receiver device would use 'WatchNextResponse data' to retrieve a media item" (Bhatta. Rebuttal Report, ¶298), Dr. Bhattacharjee contends:

> But Dr. Schmidt misunderstands this alternative. Google's documents indicate that it has discussed different options for implementing Streaming Watch. For playing back playlists on a receiver device, Streaming Watch ***could be*** implemented such that the Onesie agent (and not the receiver) would use the playlistId to identify the starting ***and subsequent media items*** in the playlist and stream back the media content to the receiver device. *See* GOOG-SONOSNDCA-00070863 (Streaming Watch (Everywhere?)) at -913 ("GetWatchRequest could just include a playlist ID. The WatchServer could determine the starting video from the playlist ID then pass that starting video (as well as other context) to the WatchUiServer."). A receiver device would not use a videoId ("WatchNextResponse data") to retrieve the content for ***the media items*** in the receiver device.

Id.

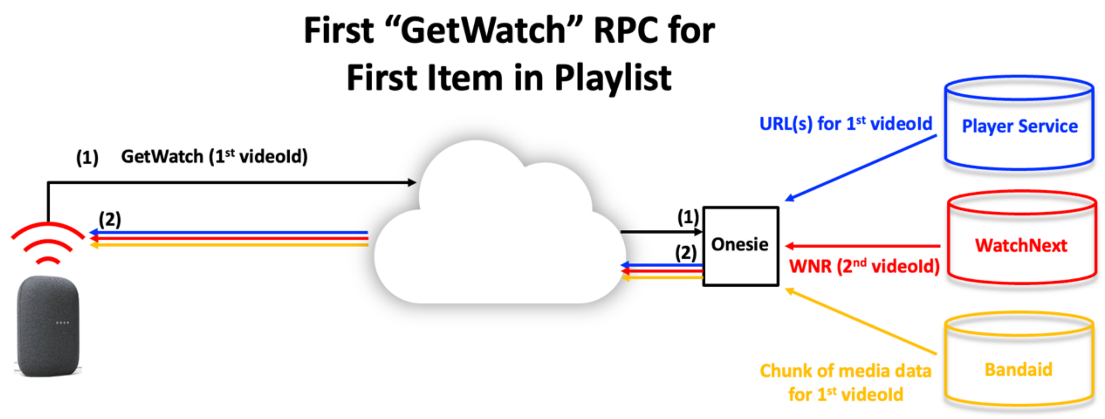
273. Dr. Bhattacharjee's explanation here is inconsistent with his prior explanation from his Showdown Report and different from what Google advanced in its interrogatory response. In this regard, Dr. Bhattacharjee is now inconsistently saying a ***single*** request to the Onesie agent would somehow result in the Receiver obtaining ***all*** subsequent media items and WatchNextResponses. In fact, Dr. Bhattacharjee's assertion is not even supported by the document

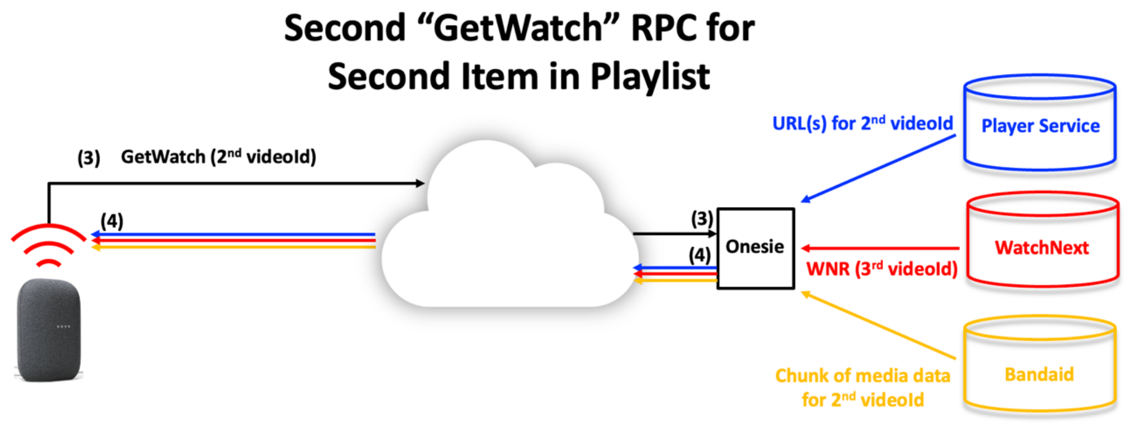
that he cites, which merely says the request that might include a playlistId results in determining the ***first*** starting video ***singular***.

274. Dr. Bhattacharjee appears to be saying that his new modification to the purported combined Onesie/Streaming-Watch alternative would involve a Onesie agent essentially becoming a streaming content provider for ***each*** Receiver that is in a Cast session in the United States at any given time, where that newly modified Onesie agent not only provides all media items to a particular Receiver but also manages its particular Receiver's playback through the remote playback queue and somehow manages all WatchNext request/response interactions on behalf of the Receiver. What's more, Dr. Bhattacharjee provides no explanation about how his new modification would enable a user to skip to the next media item without the Receiver storing the videoId of that next media item and exchanging it with the Onesie agent. In this way, Dr. Bhattacharjee appears to be proposing to drastically alter the workload and sophistication of each Bandaid server serving as a Onesie agent and again, there would need to be a newly-modified Onesie agent for each Receiver that is in a Cast session in the United States at any given time. Simply put, it is my opinion that such a proposed design would not have been technically feasible and Dr. Bhattacharjee has certainly not explained how it could have been.

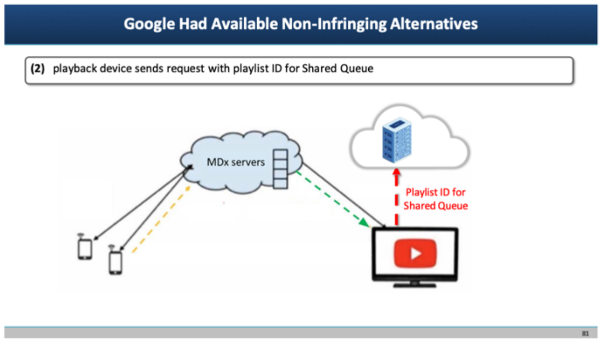
275. Regardless, I understand that, because Google's interrogatory response incorporates Dr. Bhattacharjee's Showdown Reports by reference and does not otherwise explain the details of the purported combined Onesie/Streaming-Watch alternative, the description in Dr. Bhattacharjee's Showdown Reports of this purported alternative is what controls. As I explained in my Rebuttal Report, the Onesie/Streaming-Watch alternative described in Dr. Bhattacharjee's Showdown Reports still would infringe limitation 1.7 because the Receiver would still be configured to "communicate with the cloud-based computing system in order to obtain data identifying a next one or more media items that are in the remote playback queue" by virtue of a first GetWatch RPC (e.g., that returns a WatchNextResponse containing, *inter alia*, a second videoId for a next media item for the Receiver to playback), which I have illustrated below, and would still be configured to "use the obtained data to retrieve at least one media item in the remote playback queue from the cloud-based media service" by virtue of a second GetWatch RPC (e.g., that returns (i) media data

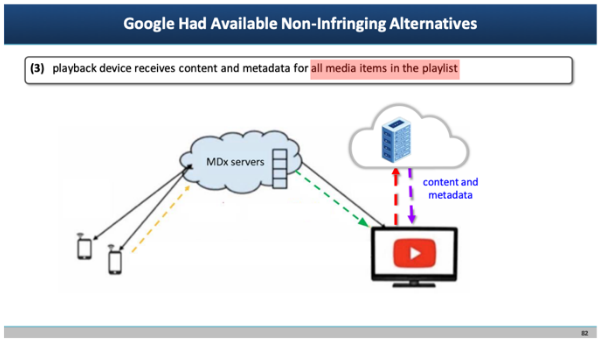
corresponding to the second videoId and (ii) a WatchNextResponse containing, *inter alia*, a third videoId for a next media item for the Receiver to playback), which I have illustrated below.



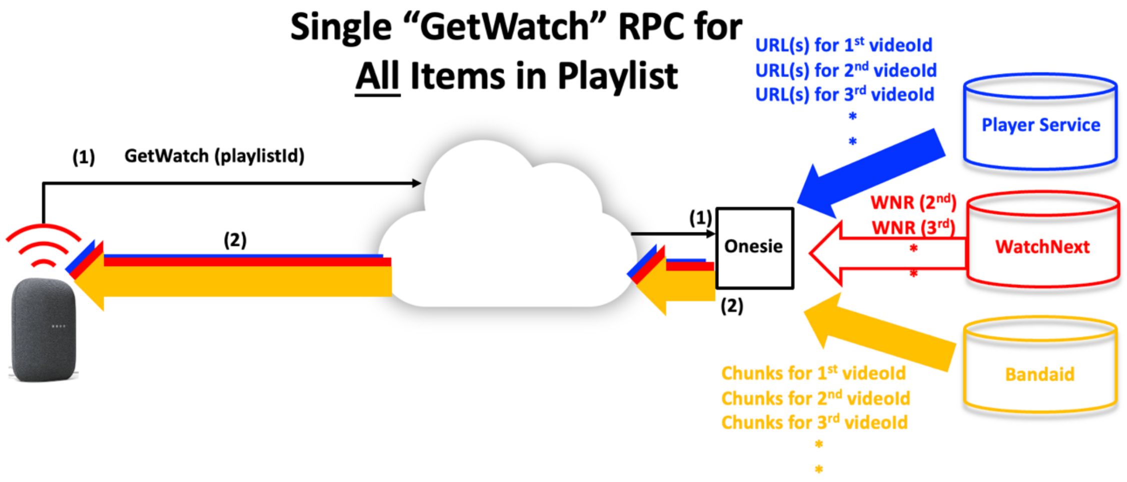


276. In stark contrast to what is illustrated above, Dr. Bhattacharjee provides two sets of illustrations of his (new) combined Onesie/Streaming-Watch alternative that show a Receiver sending a single request containing a playlistId for a given playlist to the Onesie agent and the Onesie agent in turn streaming media data and meta-data for ***all*** media items in the playlist, one set of which is reproduced below:


Google Had Available Non-Infringing Alternatives
(2) playback device sends request with playlist ID for Shared Queue
MDx servers
Playlist ID for Shared Queue
81


Google Had Available Non-Infringing Alternatives
(3) playback device receives content and metadata for all media items in the playlist
MDx servers
content and metadata
82

277. Again, Dr. Bhattacharjee's illustrations provide no hint as to how this is technically feasible and obscures the new modifications that he is proposing. The illustration below more accurately reflects some of the necessary (undiscussed and unexplained) modifications for Dr. Bhattacharjee's (new) combined Onesie/Streaming-Watch alternative:



278. Dr. Bhattacharjee attempts to minimize the testimony of Google's 30(b)(6) witness, Mr. Mo, that undermines Google and Dr. Bhattacharjee's assertion that "Google has implemented Onesie for many receiver devices and is currently working on a feature called Streaming Watch …." *See* Bhatta. Rebuttal Report, ¶290. But I find his positions to be flawed.

279. **First**, Dr. Bhattacharjee acknowledges that Mr. Mo was designated Google's 30(b)(6) witness for "the accused casting functionality." *Id.*, ¶291. The purported Onesie/Streaming-Watch alternative of course relates to "the accused casting functionality." *See, e.g.*, Bhatta. Op. Showdown Report, ¶590 ("And although I understand that Onesie has not yet been implemented ***for casting***, I understand that Google anticipates that it will be released ***for the casting***

Dated: January 23, 2023

Douglas C. Schmidt

DOUGLAS C. SCHMIDT